This listing of claims will replace all prior versions, and listings, of claims in the application:

The Status of the Claims

1. (Currently Amended) A method to protect a protocol interface comprising:

receiving a driver request from a driver during an operation phase of firmware in a processor system;

identifying the driver request as a request associated with a violating condition of the protocol interface; and

rejecting the driver request.

2. (Original) A method as defined in claim 1, wherein receiving the driver request during the operating phase of firmware comprises receiving the driver request during one of a pre-EFI initialization (PEI) phase and a driver execution environment (DXE) phase.

3. (Original) A method as defined in claim 1, wherein identifying the driver request as the request associated with the violating condition of the protocol interface comprises identifying at least one of a request to access an architectural protocol installed in the processor system, a reinstall request, and an install request by a driver.

4. (Original) A method as defined in claim 1, wherein identifying the driver request as a request associated with the violating condition of the protocol interface comprises identifying the driver request associated with a violating condition of a central processing unit (CPU) architectural protocol.

5. (Original) A method as defined in claim 1, wherein rejecting the driver request comprises storing the protocol interface in a data structure in response to identifying a request by a driver to access an architectural protocol installed in the processor system.

6. (Original) A method as defined in claim 1, wherein rejecting the driver request comprises rejecting one of a reinstall request and an install request by a driver.

7. (Original) A machine readable medium storing instructions, which when executed, cause a machine to:

receive a driver request during an operation phase of firmware in a processor system;

identify the driver request as a request associated with a violating condition of a protocol interface; and

reject the driver request.

8. (Original) A machine readable medium as defined in claim 7, wherein the instructions cause the machine to receive the driver request during an operating phase of firmware by receiving the driver request during one of a pre-EFI initialization (PEI) phase and a driver execution environment (DXE) phase.

9. (Original) A machine readable medium as defined in claim 7, wherein the instructions cause the machine to identify the driver request associated with the violation condition of the protocol interface by identifying at least one of a request to access an architectural protocol installed in the processor system, a reinstall request, and an install request by a driver.

10. (Original) A machine readable medium as defined in claim 7, wherein the instructions cause the machine to identify the driver request associated with the violation condition of the protocol interface by identifying the driver request associated with a violating condition of a central processing unit (CPU) architectural protocol.

11. (Original) A machine readable medium as defined in claim 7, wherein the instructions cause the machine to reject the driver request by storing the protocol interface in a data structure in response to identifying a request by a driver to access an architectural protocol installed in the processor system.

12. (Original) A machine readable medium as defined in claim 7, wherein the instructions cause the machine to reject the driver request by rejecting one of a reinstall request and an install request by a driver.

13. (Original) A machine readable medium as defined in claim 7, wherein the machine readable medium comprises one of a programmable gate array, application specific integrated circuit, erasable programmable read only memory, read only memory, random access memory, magnetic media, and optical media.

14. (Currently Amended) An apparatus to protect a protocol interface comprising:

a data structure configured to store one or more protocol interfaces;

a processor operatively coupled to the data structure and to a machine readable medium storing instructions that, when executed, cause the processor, the processor being programmed to receive a driver request from a driver during an operation phase of firmware in a processor system, to identify the driver request as a request associated with a violating condition of the protocol interface, and to reject the driver request.

15. (Original) An apparatus as defined in claim 14, wherein the driver request comprises one of a request to access an architectural protocol installed in the processor system, a reinstall request, and an install request by the driver.

16. (Original) An apparatus as defined in claim 14, wherein the protocol interface comprises a central processing unit (CPU) architectural protocol.

17. (Original) An apparatus as defined in claim 14, wherein the operation phase comprises one of a pre-EFI initialization (PEI) phase and a driver execution environment (DXE) phase.

18. (Original) An apparatus as defined in claim 14, wherein the processor is configured to store the protocol interface in the data structure in response to identifying a request by a driver to access an architectural protocol installed in the processor system.

19. (Currently Amended) A processor system to protect a protocol interface comprising:

a dynamic random access memory (DRAM) configured to store one or more protocol interfaces;

a processor operatively coupled to the data structure DRAM and to a machine readable medium storing instructions that, when executed, cause the processor, the processor being programmed to receive a driver request from a driver during an operation phase of firmware in the processor system, to identify the driver request as a request associated with a violating condition of the protocol interface by the driver, and to reject the driver request.

20. (Original) A processor system as defined in claim 19, wherein the driver request comprises one of a request to access an architectural protocol installed in the processor system, a reinstall request, and an install request by the driver.

21. (Original) A processor system as defined in claim 19, wherein the protocol interface comprises a central processing unit (CPU) architectural protocol.

22. (Original) A processor system as defined in claim 19, wherein the operation phase comprises one of a pre-EFI initialization (PEI) phase and a driver execution environment (DXE) phase.

23. (Original) A processor system as defined in claim 19, wherein the processor is configured to store the protocol interface in the data structure in response to identifying a request by a driver to access an architectural protocol installed in the processor system.